上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

在线协作文档

最终迭代汇报展示

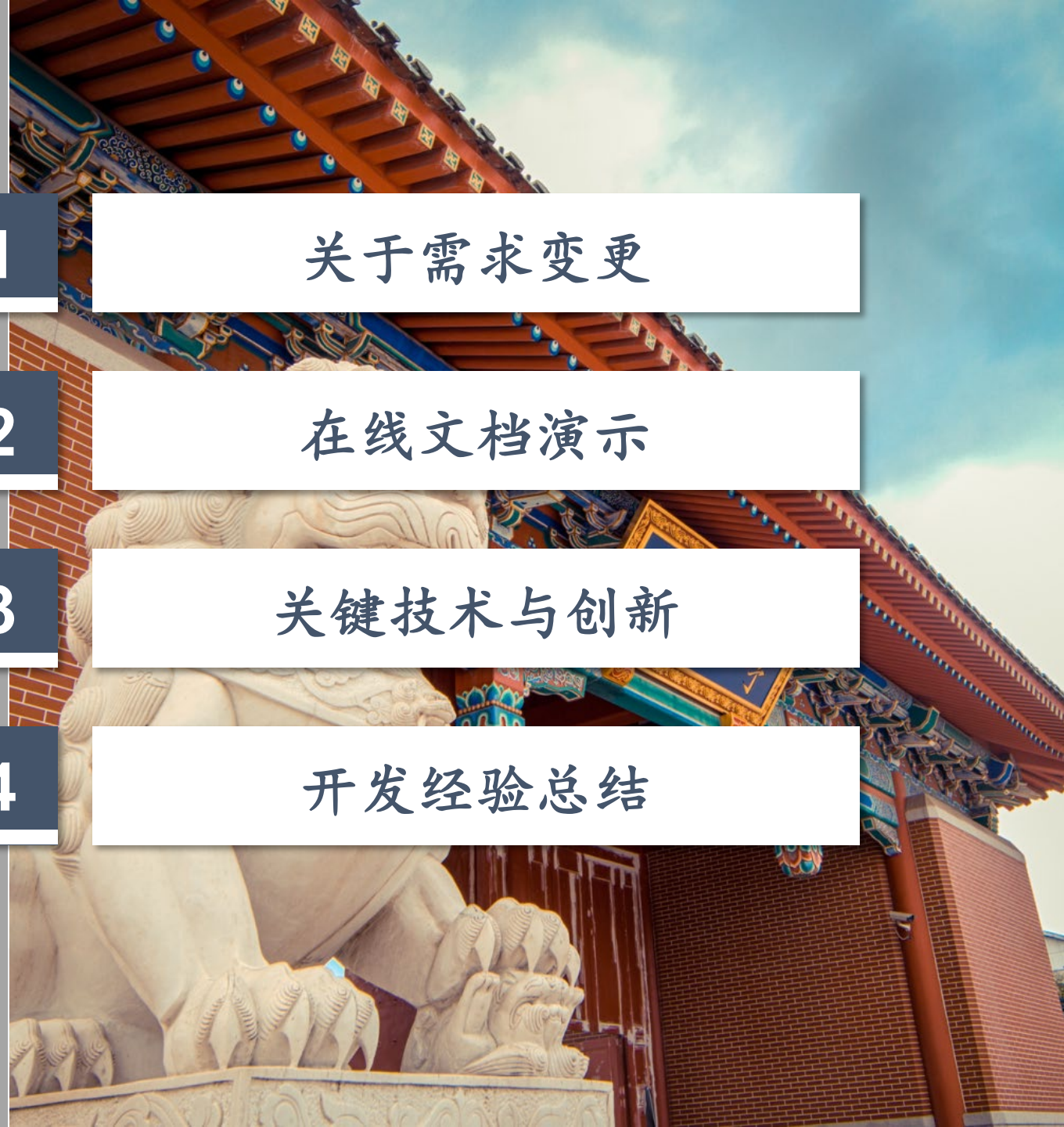
小组成员：李思旷、张子谦、刘骏霖、姜凯

饮水思源 • 爱国荣校

SJTU

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

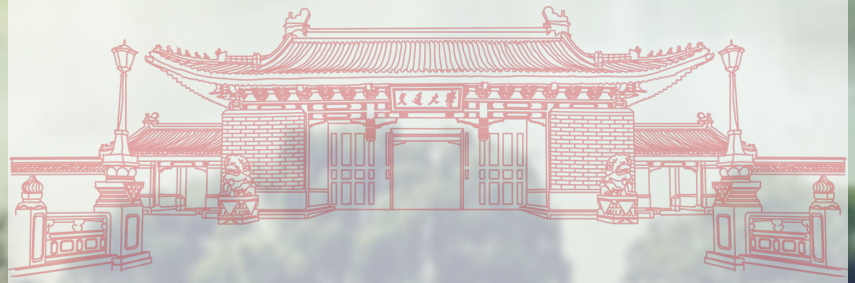
目录



01

关于需求变更

Requirements Change



在开发中出现的分歧

坚持开发足迹

- 已设计完成了的基本界面UI。
- 开发文档的编写基于足迹App。
- 微信小程序轻量级特点。
- 变更需求的成本较大。

开发在线文档

- ✓ 具有更多已有的库支持。
- ✓ 更多的学习样例资源参考。
- ✓ 小组成员缺乏移动端开发经验。
- ✓ 本学期的课程压力较大。
- ✓ 相对学习成本低。可借鉴本学期互联网应用开发技术的知识，快速实现需求。

关于需求变更

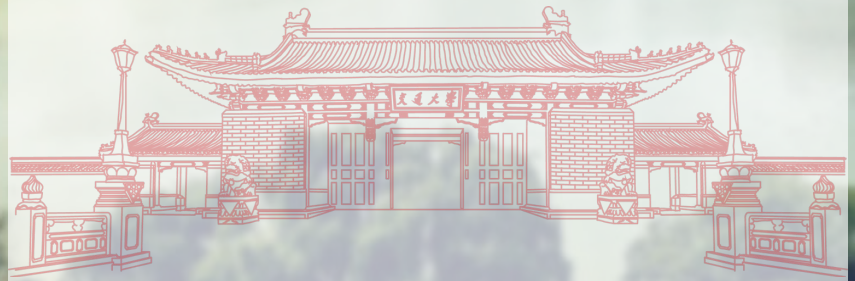
在考虑变更选题后？

目前也已经实现了基本的功能，完成本学期的迭代的目标。

02

在线文档演示

Effect Display



产品背景简介

众所周知，交大目前已有的在线文档有两个，一个是水源文档，一个是“Overleaf”，但是我们在日常的使用过程中发现了很多不足：

用户管理？No！（没有用户的管理机制）

附件上传？No！（只能上传图片）

附件管理？No！（没有对附件的管理功能）

电子表格？No！（语法仅限Markdown）

团队管理？No！（只能通过分享链接）

产品部署简介

- ✓服务器已部署于交大云主机，可以在校园网内部访问。
- ✓假定面向用户为交大师生，设计界面具有交大的特色。
- ✓保留了旧版的水源文档服务，并且已部署到云主机上。
- ✓旧版的水源文档服务供有极高实时性要求的用户使用。
- ✓支持单点登录、预留了一定的配置空间。（功能测试）

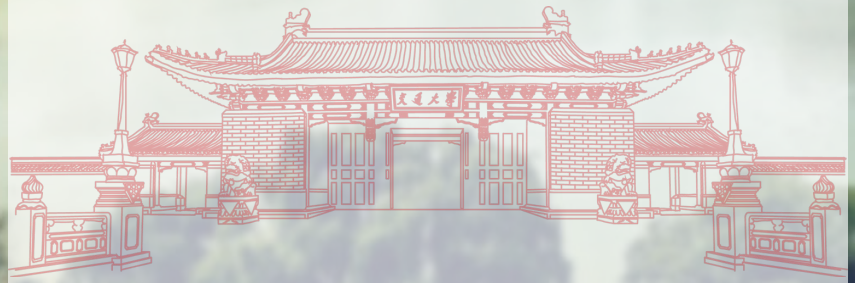
产品特色与创新点

- ✓支持Markdown基本文档的编写。
- ✓支持基本思维导图的渲染与展示。
- ✓支持Latex语法的数学公式渲染。
- ✓支持在线编辑电子表格的内容。
- ✓支持图片视频等各种附件的上传，并保存在云服务器上。
- ✓最大单次上传数据约为10GB，当然可以用户个性配置。
- ✓支持框架的嵌入，可以在文档中播放视频、查看PDF。
- ✓用户协作的编辑的页面已经设计完成，已经与后端接口对接。

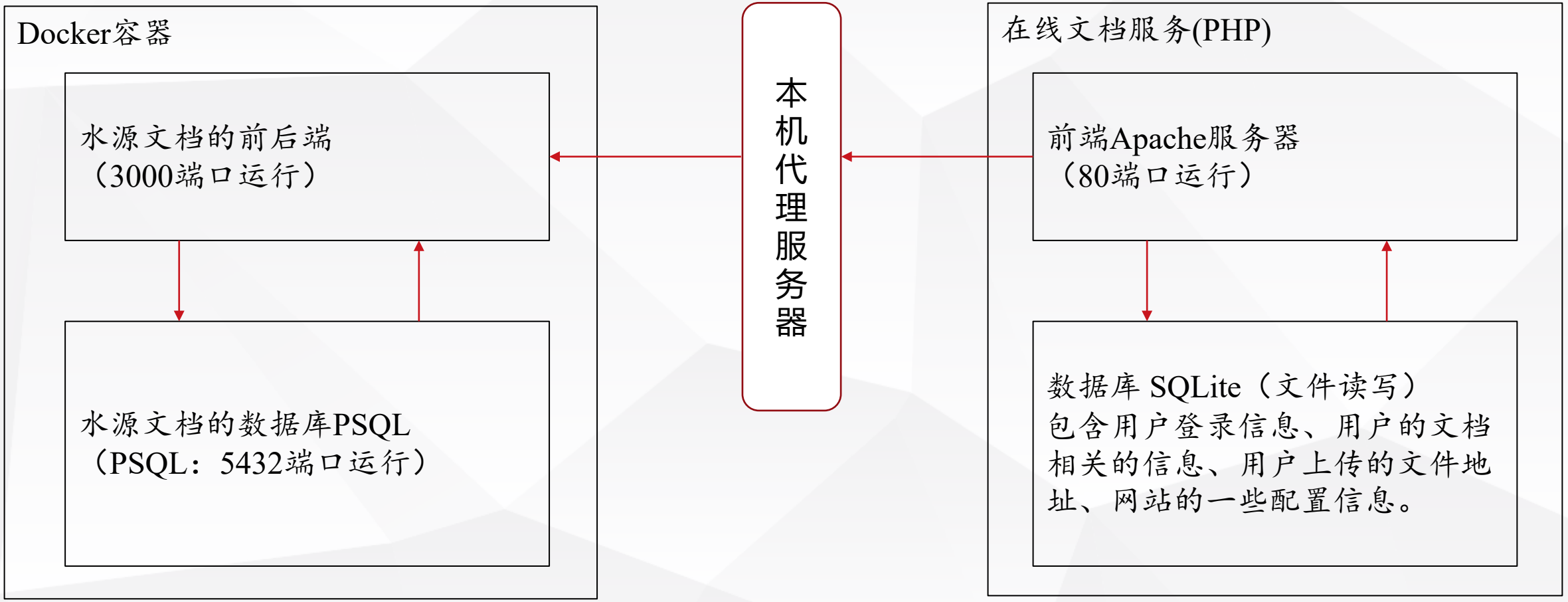
03

技术创新简介

Tech Introduction

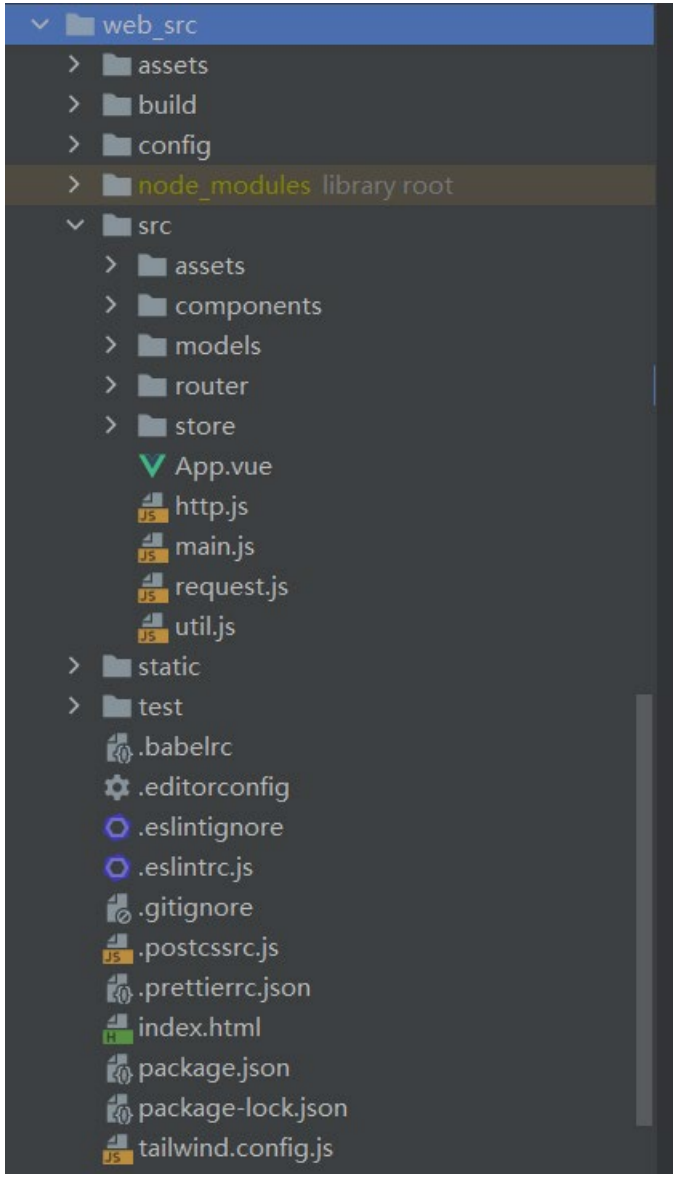


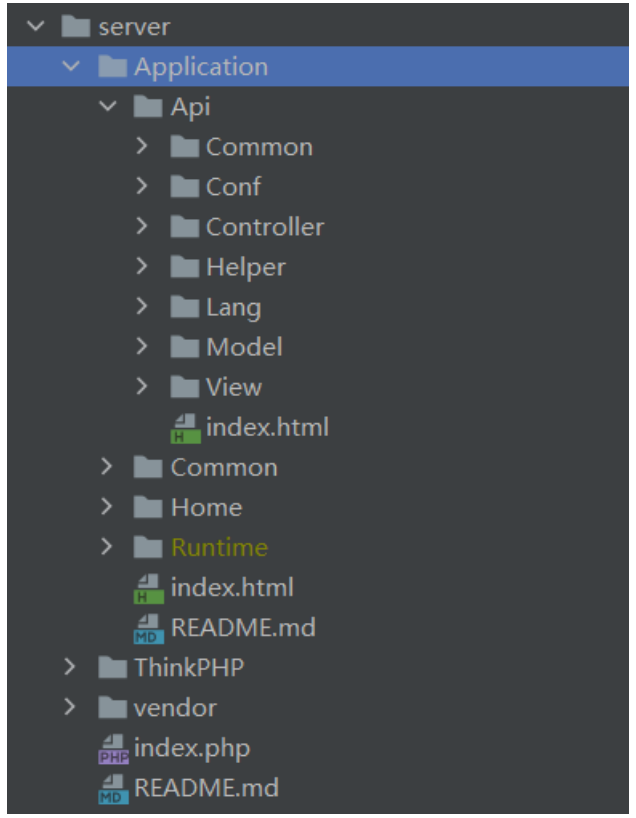
部署架构


Docker容器
水源文档的前后端
（3000端口运行）
水源文档的数据库PSQL
（PSQL：5432端口运行）
本机代理服务器
在线文档服务(PHP)
前端Apache服务器
（80端口运行）
数据库 SQLite（文件读写）
包含用户登录信息、用户的文档相关的信息、用户上传的文件地址、网站的一些配置信息。

开发简介

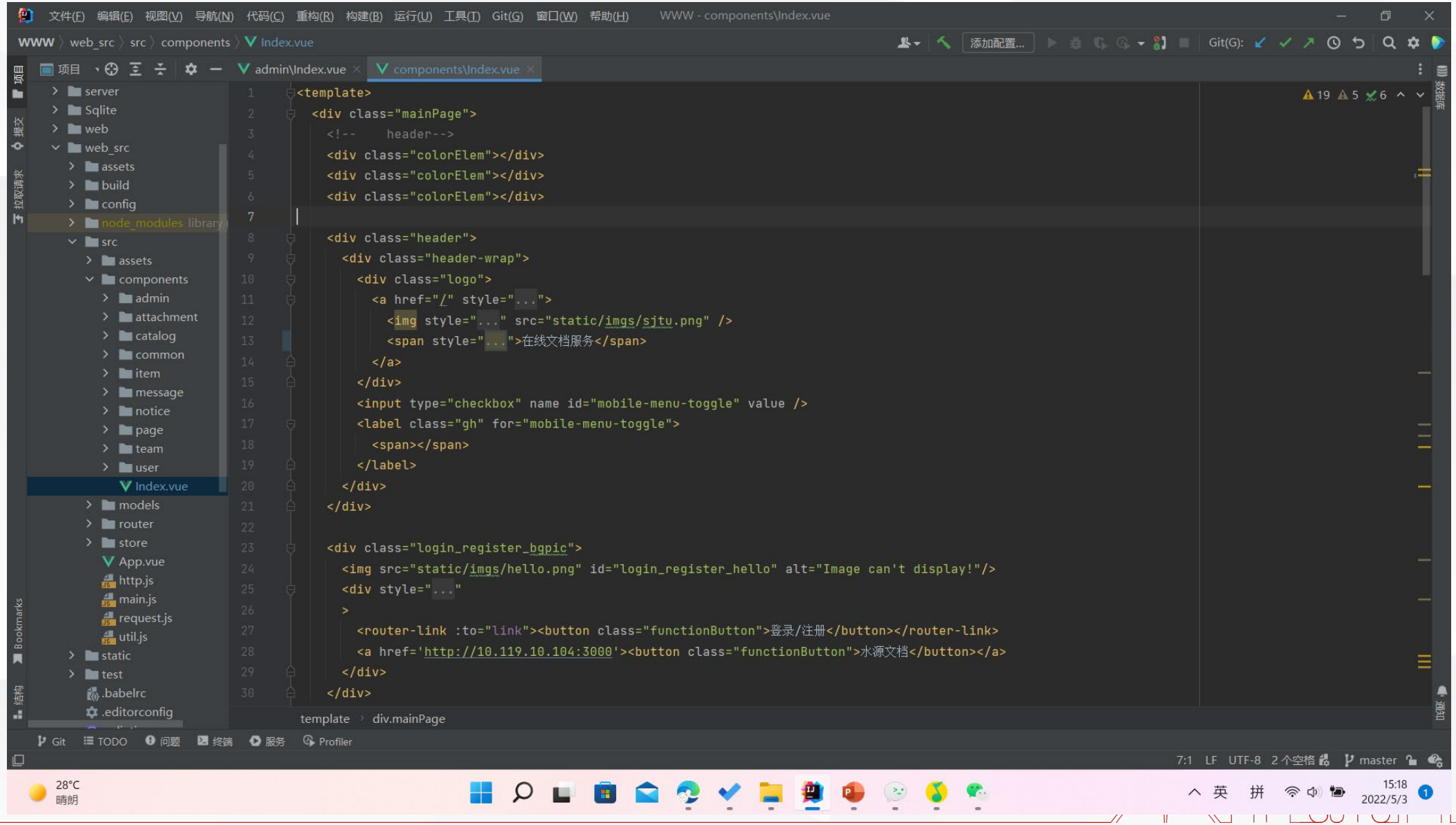
- ✓整体开发基于前后端分离的思想，前端的部分部署在 Apache 服务器中，响应 80 端口的Http/Https请求。
- ✓前端开发基于 Vue 框架和 Element-UI，使用了一些组件库，运用了组件化开发的思想，增强的组件的复用性。
- ✓后端的开发基于 PHP 环境，技术仍在学习阶段，学习借鉴了一些已有的开发样例。
- ✓后端业务逻辑分层，API部分暴露在外面，Controller 层负责处理前端发送来的各种请求，校验是否权限是否合法。合理处理请求将会发送到Model进行实际操作。
- ✓Model提供了一些实际操作的模型，对接数据库，进行增删改查的基本操作。例如用户可能有的操作包括：数据库中增加用户、数据库中删除用户、修改用户的密码等相关的业务。


web_src
assets
build
config
node_modules library root
src
assets
components
models
router
store
App.vue
http.js
main.js
request.js
util.js
static
test
.babelrc
.editorconfig
.eslintignore
.eslintrc.js
.gitignore
.postcssrc.js
.prettierrc.json
index.html
package.json
package-lock.json
tailwind.config.js

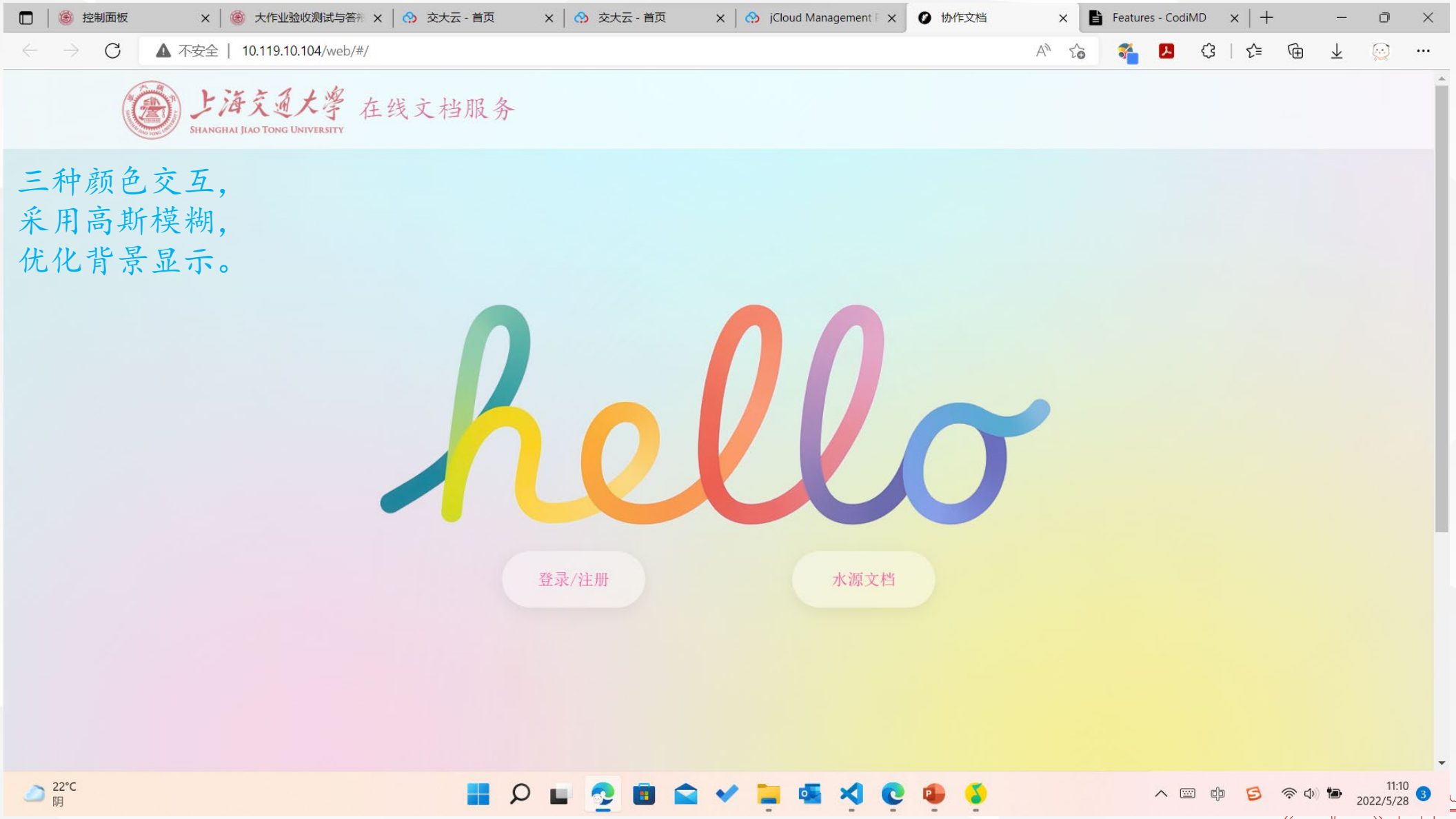

server
Application
Api
Common
Conf
Controller
Helper
Lang
Model
View
index.html
Common
Home
Runtime
index.html
README.md
ThinkPHP
vendor
index.php
README.md

部分代码展示-前端

三种颜色交互，
采用高斯模糊，
优化背景显示。



显示效果展示-前端


控制面板
大作业验收测试与答辩
交大云 - 首页
交大云 - 首页
jCloud Management
协作文档
Features - CodiMD
不安全 | 10.119.10.104/web/#/
上海交通大学
SHANGHAI JIAO TONG UNIVERSITY
在线文档服务
三种颜色交互，
采用高斯模糊，
优化背景显示。
hello
登录/注册
水源文档
22°C
阴
11:10
2022/5/28

部分代码展示-后端Controller

提供API给外部接口的函数

components\Index.vue | index.php | UserModel.class.php | UserController.class.php | MessageContentModel.class.php | ItemTokenMo

```php
class UserController extends BaseController
{


    //注册
    public function register(){...}

    //导入示例项目
    private function _importSample($uid){...}

    private function _importZip($file, $uid){...}

    //登录
    public function login(){...}

    //登录2
    public function loginByVerify(){...}

    //注册2
    public function registerByVerify(){...}

    //获取用户信息
    public function info(){...}

    //获取所有用户名
    public function allUser(){...}

    //通过旧密码验证来更新用户密码
    public function resetPassword(){...}
```

部分代码展示-后端Model

用户可能用到的一些操作逻辑：
例如注册时要判断是否有已经存在的用户，注册用户要进行添加数据，修改密码时要更新数据库。

```php
class UserModel extends BaseModel
{

    /**
     * 用户名是否已经存在
     *
     */
    public function isExist($username)
    {
        return  $this->where( where: "username = '%s'", array($username))->find();
    }

    /**
     * 注册新用户
     *
     */
    public function register($username, $password)
    {
        $salt = get_rand_str();
        $password = encry_password($password, $salt);
        $uid = $this->add(array('username' => $username, 'password' => $password, 'salt' => $salt,  'reg_time' => time()));
        return $uid;
    }

    //修改用户密码
    public function updatePwd($uid, $password)
    {
        $res = $this->where( where: "uid = '%d'", array($uid))->find();
        $password = encry_password($password, $res['salt']);
        return $this->where( where: "uid ='%d' ", array($uid))->save(array('password' => $password));
```

One More Thing

没有创新点？

One More Thing……

单点登录的尝试

SSO？

OAuth 2.0？

OIDC？

OpenID是用来认证的，就是说可以用一个url来唯一表明身份（不用挨个记每个网站的用户密码）。OAuth是用来授权的（可以授权一个网站访问公司/集体在另外一个网站（资源服务器）的数据，而不用把密码给第一个网站，通过临时的accessToken

OAuth 2的尝试

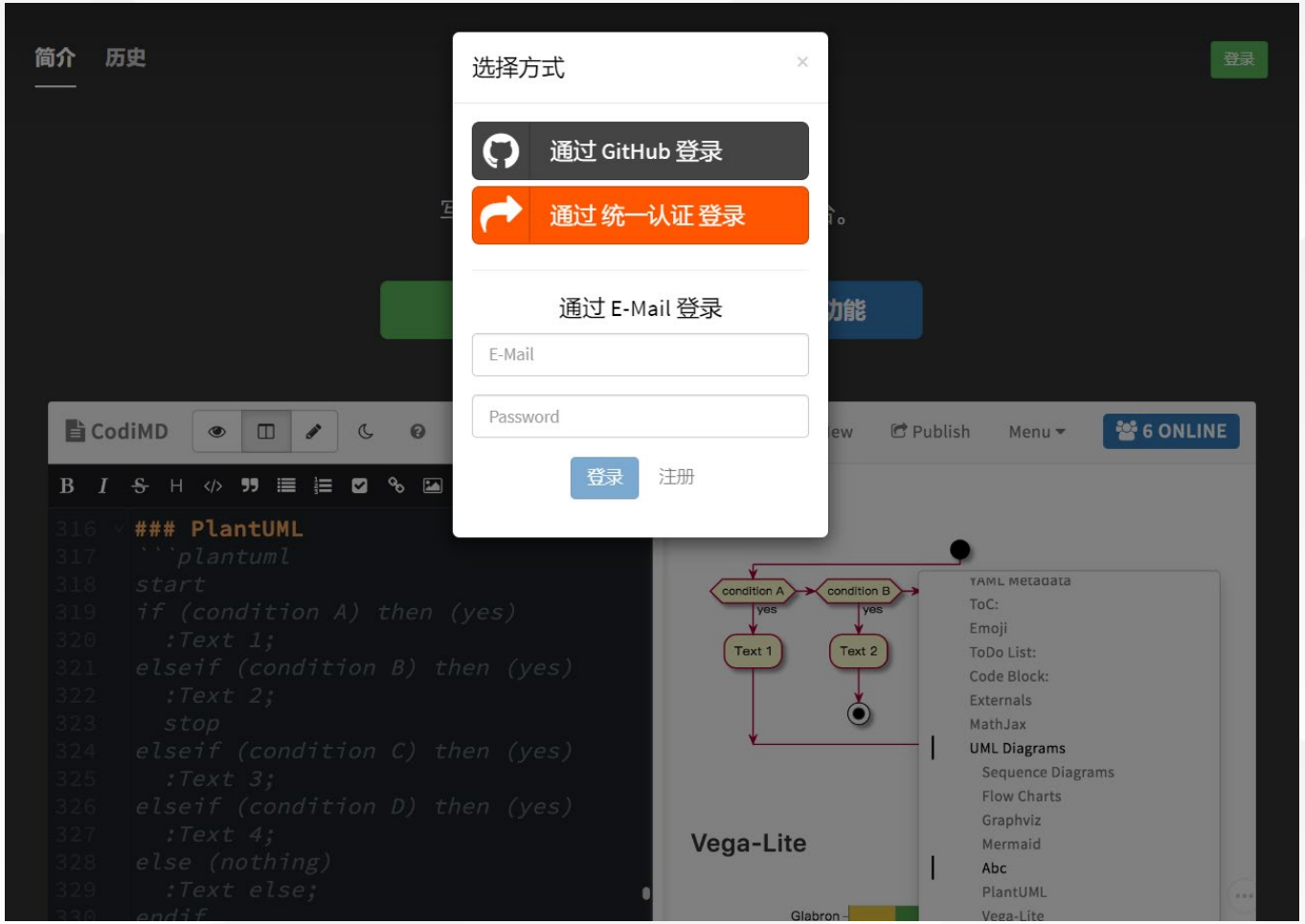
在线文档支持社区的登录页面：

理想的单点登录效果：
支持在我们的在线文档、原有的水源文档和支持服务社区之间自如切换。

全线单点登录已支持！同时交付三个APP

通过 Jaccount 是否可行？
理论可行，但是需要向学校申请，得到开发者的client_ID和secret_ID，本次作为大作业就没有使用。

OAuth 2的尝试



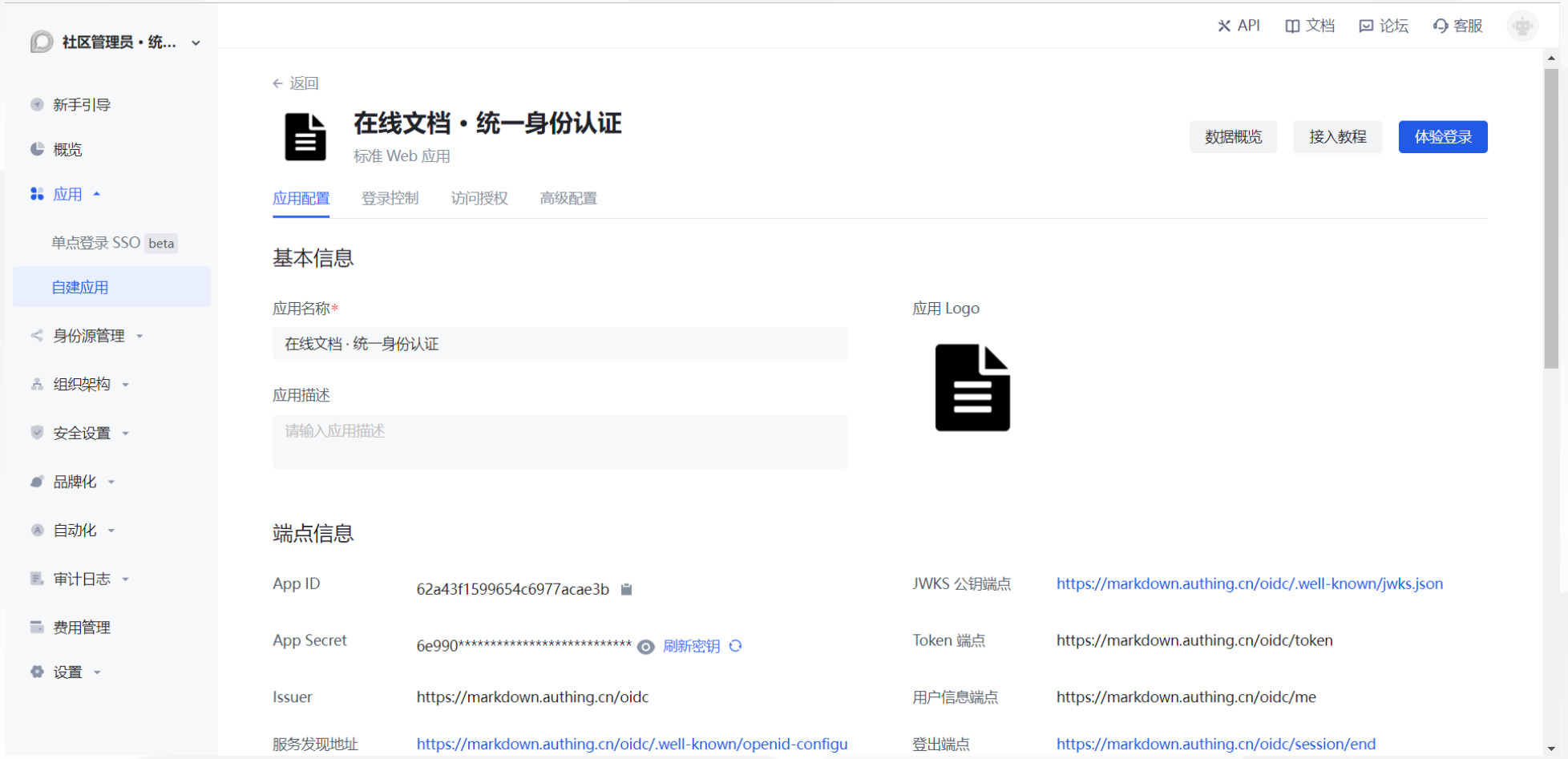
水源文档的登录效果配置已完成。

理想的单点登录效果：
支持在我们的在线文档、原有的水源文档和支持服务社区之间自如切换。

用户名ssouser
密码123456

通过 Jaccount 是否可行?
理论可行，但是需要向学校申请，完成一系列的流程，得到开发者的client_ID、secret_ID，本次作为大作业就没有使用。

OAuth 2的尝试（授权码模式）



中央认证服务器配置页面

OAuth 2的尝试（授权码模式）

1、请求登录授权
类比在教务网单击通过Jaccount登录

例如：
https://i.sjtu.edu.cn/jaccountlogin

例如某次单击登录跳转后的链接是：
https://jaccount.sjtu.edu.cn/jaccount/jalogin?sid=jaoauth220160718&client=CI2Y6cp96NGE7khPOJZlEzw3CUCQ5Sa%2Fi3o5%2Bki8%2BaEe&returl=CKNPF7GChF%2BnHinZ2iwM9bhVFyoF2SDecAabGZxDV3pRqoQ1evINDaek%2BqR97P52vrakTNzlH9e4tsryB0Pq48Tl3nRyZ1ahEP9MUZgDV%2BVMkV2oSedMF2qTfRvcjxQrNw%3D%3D&se=CNUylFo58VApN5qqRjuTCLL2opc5HrRsbX%2FR75x4oDYst99kKYA%2BBDfbDpiWptd0nXaz3eMSKU6p

OAuth 2的尝试（授权码模式）

1、请求登录授权
类比在水源文档单击通过统一认证登录

https://markdown.authing.cn/login?app_id=62a43f1599654c6977acae3b&uuid=0AaS29YNhQIIO19lDwkJE&finish_login_url=%2Finteraction%2Foidc%2F0AaS29YNhQIIO19lDwkJE%2Flogin&tenant_id=

OAuth 2的尝试（授权码模式）

2、登录回调（对应图中的四）
中央认证服务器在用户输入了正确的用户名密码之后，会回调下面的url，具体说来表现为，浏览器的地址栏发生变化，例如变成下面的结果：

https://<markdown.myapp.com>/auth/oauth2/callback?code=XXXXXXXX&&state=XXXXXXXX

参数必须要附带上code，这个是授权码。当然也可以带上state，这个是表示时效性的一个状态，用于服务器校验。当然具体根据app而异。

根据 Jaccount 的开发者文档解释：
state：当应用请求时发送了 state 参数时，授权服务器**必须**返回该参数。
授权码是一次性的！

OAuth 2的尝试（授权码模式）

2、后端处理回调（对应图中的5-8）

接下来的操作在后端完成，用户不可见

服务器接到了Get请求，解析Get请求里面的参数，获得授权码。
获得授权码之后，利用code，向中央认证服务器发送请求，得到access_Token。**这个令牌是用户访问资源服务器的临时凭证。**

中央认证服务器接收access_Token的端点是(以我自部署的水源文档为例):
https://<markdown.authing.cn>/oidc/token

OAuth 2的尝试（授权码模式）

2、后端处理回调（对应图中的5-8）

操作在后端完成，用户不可见

应用服务器接到access_Token之后，再利用AccessToken去换取用户信息，得到用户的一个json字符串。

中央认证服务器和资源服务器往往合二为一，这一步骤获取用户信息的的端点是(以我自部署的水源文档为例):
https://<markdown.authing.cn>/oidc/me

例如Jaccount的对应地址是
https://api.sjtu.edu.cn/v1/me/profile

OAuth 2的尝试（遇到的BUG）

错误类型（一）：

用户在统一身份认证输入正确的用户名密码的之后，网页跳转，然后报错：

500服务器内部错误（Server Internal Error）

错误步骤在4-8

常见的类型错误：

1、Get Post请求错误，OAuth2仅仅是一个协议的约束，没有对每一步如何请求（Get还是Post请求类型）做一个约束

2、用户信息JSON中字段没有识别。例如水源文档识别的是SUB，而不是 User 也不是 Name，也不是UserName

错误类型（二）：

无法打开第三方的单点登录页面：

常见的类型错误：

ClientID和SecretID异常配置，需要检查。

OAuth 2的疑问（思考的问题）

1、正如刚才所述，授权码暴露给用户了，是否存在风险？

没有，即使被截获，由于没有后端的ClientID，SecretID第三方无法获取accessToken。
Client_ID，Secret_ID在后端，特别注意Secret_ID不能泄露。否者会很危险。
也就是说，这样可以保证，只有后端的服务器，可以获取到资源，然后前台用户得到显示的登录成功的效果。避免非法获得用户信息。

确保直接获得用户资源信息的唯一方式是：应用服务器后端
2、为什么要用code换token，用token的accessToken，再换得用户资源信息？

Code有效期短（用于登录瞬间的认证，让应用服务器知道刚刚是谁登录了），accessToken有效期长，用于用户会话期间资源请求。

OAuth 2的尝试（遇到的BUG）

错误类型（一）：

用户在统一身份认证输入正确的用户名密码的之后，网页跳转，然后报错：

500服务器内部错误（Server Internal Error）

错误步骤在4-8

常见的类型错误：

1、Get Post请求错误，OAuth2仅仅是一个协议的约束，没有对每一步如何请求（Get还是Post请求类型）做一个约束

2、用户信息JSON中字段没有识别。例如水源文档识别的是SUB，而不是 User 也不是 Name，也不是UserName

错误类型（二）：

无法打开第三方的单点登录页面：

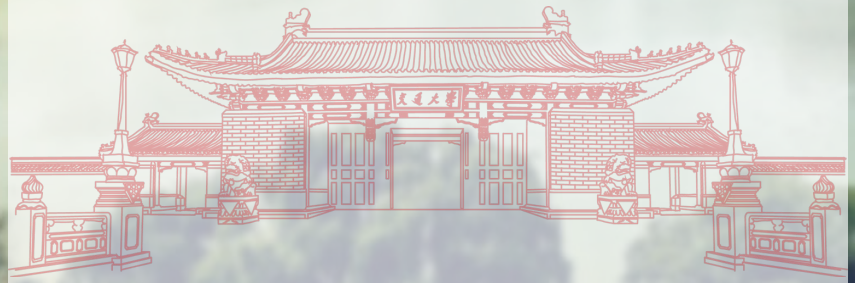
常见的类型错误：

ClientID和SecretID异常配置，需要检查。

04

开发经验总结

Experiences Of All



开发经验总结

1、实事求是，量力而行、拒绝好高骛远。

2、合理的学习已有的代码，调用第三方的库文件。

3、小组成员内的及时沟通。

感谢聆听

饮 水 思 源　爱 国 荣 校